Freeform Search

Database: US Pre-Grant Publication Full-Text Database; US Patents Full-Text Database; US OCR Full-Text Database; EPO Abstracts Database; JPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Term: L6 and (scatter$ near list)

Display: 100 **Documents in Display Format:** - **Starting with Number** 1

Generate: ○ **Hit List** ◉ **Hit Count** ○ **Side by Side** ○ **Image**

Search Clear Interrupt

Search History

DATE: Tuesday, January 17, 2006 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L7	L6 and (scatter$ near list)	10	L7
L6	L5 and scatter$	50	L6
L5	L4 and (data near element)	488	L5
L4	L2 and (memory near location)	3494	L4
L3	L2 and (memoy near location)	2	L3
L2	TRANSFER$ NEAR DATA NEAR MEMOR$	12729	L2
L1	TRANSFER NEAR DATA NEAR MEMOR$	7940	L1

END OF SEARCH HISTORY

L7: Entry 5 of 10 File: USPT Jan 7, 2003

DOCUMENT-IDENTIFIER: US 6505268 B1
TITLE: Data distribution in a disk array

Detailed Description Text (6):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (8):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (26):
A request block 204 is comprised of two parts, a fixed length request header 206 and variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptors which define data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/C structures. The request header is a total of 12 bytes in length.

Detailed Description Text (27):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Thus, the present invention, permits a read command to read data from two distinct, non-contiguous addresses in either system or disk memory. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the request blocks 204 in the command list, the scatter/gather descriptors 208 must be contiguous and, if there exists a second scatter/gather descriptor 208 set for a request, it must directly follow the first set of scatter/gather descriptors 208.

Detailed Description Text (32):
The present invention is directed toward a method for distributing data among the disk array drives for I/O commands, such as READ or WRITE. The I/O commands instruct the disk array

controller 112 to perform scatter/gather operations on sequential blocks of data. This scatter/gather descriptor structure is used by the disk array controller 112 to locate data within the array. The descriptor structure may specify buffer addresses and buffer lengths for data to be transferred to or from system memory 58. The total buffer length must equal the number bytes to be transferred for any I/O operation.

Detailed Description Text (33):
The read command transfers sequential blocks of data from the disk into buffers in the system memory 58. Two scatter/gather descriptors are available to specify dual destinations for the data. The prereferred embodiment also includes a method for specifying partial block transfers. If an initial buffer address is OFFFFFFFh (NULL), the preferred embodiment will skip to the offset of the requested bytes and the data for the specified is effectively ignored. The present invention will then read the remainder of the data within the particular block and transfer it to the address as requested. A null address will generate an error during a write operation.

Detailed Description Text (34):
The write command transfers data from the system memory 58 or device driver and writes it to sequential blocks on the disk drive array. A scatter/gather descriptor count number of 2 is ignored by a write command. The diagnostic command is a special command in the preferred embodiment that allows the direct manipulation of hardware. This command is generally issued as the only request in a command list. The only valid field in a diagnostic command is the command field. If there exist any outstanding request when the diagnostic command is submitted, an abort error will be returned. Once the disk array controller has been placed in a diagnostic mode, the disk array controller 112 is ready to accept diagnostic commands upon receipt of the command complete notification. The disk array controller 112 will remain in diagnostic mode until otherwise notified and will not process nondiagnostic command list.

Detailed Description Text (64):
The present invention is directed to accepting a complete logical command (including scatter/gather descriptors) and translating the logical commands to the physical operations necessary to fulfill the logical command. This translation process includes converting the original scatter/gather descriptions into drive specific information that accompanies each physical command. The translation process is based on the selected controller configuration, which takes into account the divisions of drives within the array into groups, the physical characteristics of the drives within each group within the logical unit, the selected error correction mode for the particular drive group, the selected parameters for the drive group, and the error history of the drive group. What follows is the method by which the present invention selects the mapping scheme to be used in distributing or gathering the data. The preferred embodiment reads a field within the configuration information which specifies the distribution techniques.

Previous Doc Next Doc Go to Doc#

L7: Entry 6 of 10 File: USPT Jun 1, 1999

DOCUMENT-IDENTIFIER: US 5909691 A
TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

Detailed Description Text (25):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (27):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (45):
A request block 204 is comprised of two parts, a fixed length request header 206 and variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptors which define data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/G structures. The request header is a total of 12 bytes in length.

Detailed Description Text (46):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Thus, the present invention, permits a read command to read data from two distinct, non-contiguous addresses in either system or disk memory. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the request blocks 204 in the command list, the scatter/gather descriptors 208 must be contiguous and, if there exists a second scatter/gather descriptor 208 set for a request, it must directly follow the first set of scatter/gather descriptors 208.

Detailed Description Text (61):
The present invention is directed toward a method for distributing data among the disk array

drives for I/O commands, such as READ or WRITE. The I/O commands instruct the disk array controller 112 to perform scatter/gather operations on sequential blocks of data. This scatter/gather descriptor structure is used by the disk array controller 112 to locate data within the array. The descriptor structure may specify buffer addresses and buffer lengths for data to be transferred to or from system memory 58. The total buffer length must equal the number bytes to be transferred for any I/O operation.

Detailed Description Text (62):
The read command transfers sequential blocks of data from the disk into buffers in the system memory 58. Two scatter/gather descriptors are available to specify dual destinations for the data. The preferred embodiment also includes a method for specifying partial block transfers. If an initial buffer address is 0FFFFFFFh (NULL), the present invention will skip to the offset of the requested bytes and the data for the specified is effectively ignored. The preferred embodiment will then read the remainder of the data within the particular block and transfer it to the address as requested. A null address will generate an error during a write operation.

Detailed Description Text (63):
The write command transfers data from the system memory 58 or device driver and writes it to sequential blocks on the disk drive array. A scatter/gather descriptor count number of 2 is ignored by a write command. The diagnostic command is a special command in the preferred embodiment that allows the direct manipulation of hardware. This command is generally issued as the only request in a command list. The only valid field in a diagnostic command is the command field. If there exist any outstanding request when the diagnostic command is submitted, an abort error will be returned. Once the disk array controller 112 has been placed in a diagnostic mode, the disk array controller 112 is ready to accept diagnostic commands upon receipt of the command complete notification. The disk array controller 112 will remain in diagnostic mode until otherwise notified and will not process nondiagnostic command list.

Detailed Description Text (93):
The present invention is directed to accepting a complete logical command (including scatter/gather descriptors) and translating the logical commands to the physical operations necessary to fulfill the logical command. This translation process includes converting the original scatter/gather descriptions into drive specific information that accompanies each physical command. The translation process is based on the selected controller configuration, which takes into account the divisions of drives within the array into groups, the physical characteristics of the drives within each group within the logical unit, the selected error correction mode for the particular drive group, the selected parameters for the drive group, and the error history of the drive group. What follows is the method by which the present invention selects the mapping scheme to be used in distributing or gathering the data. The preferred embodiment reads a field within the configuration information which specifies the distribution techniques.

Previous Doc Next Doc Go to Doc#

L7: Entry 7 of 10 File: USPT Jan 7, 1997

DOCUMENT-IDENTIFIER: US 5592648 A
TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

Detailed Description Text (25):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (27):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (45):
A request block 204 is comprised of two parts, a fixed length request header 206 and variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptors which define data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/G structures. The request header is a total of 12 bytes in length.

Detailed Description Text (46):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Thus, the present invention, permits a read command to read data from two distinct, non-contiguous addresses in either system or disk memory. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the request blocks 204 in the command list, the scatter/gather descriptors 208 must be contiguous and, if there exists a second scatter/gather descriptor 208 set for a request, it must directly follow the first set of scatter/gather descriptors 208.

Detailed Description Text (61):
The present invention is directed toward a method for distributing data among the disk array

drives for I/O commands, such as READ or WRITE. The I/O commands instruct the disk array controller 112 to perform scatter/gather operations on sequential blocks of data. This scatter/gather descriptor structure is used by the disk array controller 112 to locate data within the array. The descriptor structure may specify buffer addresses and buffer lengths for data to be transferred to or from system memory 58. The total buffer length must equal the number bytes to be transferred for any I/O operation.

Detailed Description Text (62):
The read command transfers sequential blocks of data from the disk into buffers in the system memory 58. Two scatter/gather descriptors are available to specify dual destinations for the data. The preferred embodiment also includes a method for specifying partial block transfers. If an initial buffer address is 0FFFFFFFh (NULL), the present invention will skip to the offset of the requested bytes and the data for the specified is effectively ignored. The preferred embodiment will then read the remainder of the data within the particular block and transfer it to the address as requested. A null address will generate an error during a write operation.

Detailed Description Text (63):
The write command transfers data from the system memory 58 or device driver and writes it to sequential blocks on the disk drive array. A scatter/gather descriptor count number of 2 is ignored by a write command. The diagnostic command is a special command in the preferred embodiment that allows the direct manipulation of hardware. This command is generally issued as the only request in a command list. The only valid field in a diagnostic command is the command field. If there exist any outstanding request when the diagnostic command is submitted, an abort error will be returned. Once the disk array controller 112 has been placed in a diagnostic mode, the disk array controller 112 is ready to accept diagnostic commands upon receipt of the command complete notification. The disk array controller 112 will remain in diagnostic mode until otherwise notified and will not process nondiagnostic command list.

Detailed Description Text (93):
The present invention is directed to accepting a complete logical command (including scatter/gather descriptors) and translating the logical commands to the physical operations necessary to fulfill the logical command. This translation process includes converting the original scatter/gather descriptions into drive specific information that accompanies each physical command. The translation process is based on the selected controller configuration, which takes into account the divisions of drives within the array into groups, the physical characteristics of the drives within each group within the logical unit, the selected error correction mode for the particular drive group, the selected parameters for the drive group, and the error history of the drive group. What follows is the method by which the present invention selects the mapping scheme to be used in distributing or gathering the data. The preferred embodiment reads a field within the configuration information which specifies the distribution techniques.

Previous Doc Next Doc Go to Doc#

L7: Entry 8 of 10 File: USPT Aug 8, 1995

DOCUMENT-IDENTIFIER: US 5440716 A
TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

Detailed Description Text (25):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (27):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (45):
A request block 204 is comprised of two parts, a fixed length request header 206 and variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptors which define data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/G structures. The request header is a total of 12 bytes in length.

Detailed Description Text (46):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Thus, the present invention, permits a read command to read data from two distinct, non-contiguous addresses in either system or disk memory. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the request blocks 204 in the command list, the scatter/gather descriptors 208 must be contiguous and, if there exists a second scatter/gather descriptor 208 set for a request, it must directly follow the first set of scatter/gather descriptors 208.

Detailed Description Text (61):
The present invention is directed toward a method for distributing data among the disk array

drives for I/O commands, such as READ or WRITE. The I/O commands instruct the disk array controller 112 to perform scatter/gather operations on sequential blocks of data. This scatter/gather descriptor structure is used by the disk array controller 112 to locate data within the array. The descriptor structure may specify buffer addresses and buffer lengths for data to be transferred to or from system memory 58. The total buffer length must equal the number bytes to be transferred for any I/O operation.

Detailed Description Text (62):
The read command transfers sequential blocks of data from the disk into buffers in the system memory 58. Two scatter/gather descriptors are available to specify dual destinations for the data. The preferred embodiment also includes a method for specifying partial block transfers. If an initial buffer address is 0FFFFFFFh (NULL), the preferred embodiment will skip to the offset of the requested bytes and the data for the specified is effectively ignored. The present invention will then read the remainder of the data within the particular block and transfer it to the address as requested. A null address will generate an error during a write operation.

Detailed Description Text (63):
The write command transfers data from the system memory 58 or device driver and writes it to sequential blocks on the disk drive array. A scatter/gather descriptor count number of 2 is ignored by a write command. The diagnostic command is a special command in the preferred embodiment that allows the direct manipulation of hardware. This command is generally issued as the only request in a command list. The only valid field in a diagnostic command is the command field. If there exist any outstanding request when the diagnostic command is submitted, an abort error will be returned. Once the disk array controller 112 has been placed in a diagnostic mode, the disk array controller 112 is ready to accept diagnostic commands upon receipt of the command complete notification. The disk array controller 112 will remain in diagnostic mode until otherwise notified and will not process nondiagnostic command list.

Detailed Description Text (93):
The present invention is directed to accepting a complete logical command (including scatter/gather descriptors) and translating the logical commands to the physical operations necessary to fulfill the logical command. This translation process includes converting the original scatter/gather descriptions into drive specific information that accompanies each physical command. The translation process is based on the selected controller configuration, which takes into account the divisions of drives within the array into groups, the physical characteristics of the drives within each group within the logical unit, the selected error correction mode for the particular drive group, the selected parameters for the drive group, and the error history of the drive group. What follows is the method by which the present invention selects the mapping scheme to be used in distributing or gathering the data. The preferred embodiment reads a field within the configuration information which specifies the distribution techniques.

Previous Doc Next Doc Go to Doc#

L7: Entry 9 of 10 File: USPT Sep 28, 1993

DOCUMENT-IDENTIFIER: US 5249279 A
TITLE: Method for controlling disk array operations by receiving logical disk requests and translating the requests to multiple physical disk specific commands

Detailed Description Text (16):
6. Scatter/Gather Block Transfer

Detailed Description Text (35):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (37):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (81):
A request block 204 is comprised of two parts, a fixed length request header 206 and variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptor counters which define data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/G descriptor counters. The request header is a total of 12 bytes in length.

Detailed Description Text (82):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Thus, the present invention is capable of reading data from two different areas in system memory. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the requests blocks 204 in the command list, the scatter/gather descriptors must be contiguous and, if there exists a second scatter/gather descriptor set for a request, it must directly follow the first set of scatter/gather descriptors.

Detailed Description Text (98):
The I/O commands instruct the disk array controller to perform scatter/gather operations on sequential blocks of data. This scatter/gather descriptor structure is used by the disk array controller to locate data within the array for transfer. The descriptor structure may specify buffer addresses and buffer lengths for data to be transferred to or from system memory 58. The total buffer length must equal the number bytes to be transferred for any I/O operation.

Detailed Description Text (99):
The read command transfers sequential blocks of data from the disk into buffers in the system memory 58. Two scatter/gather descriptors are available to specify dual destinations for the data. The present invention also includes a method for specifying partial block transfers. If an initial buffer address is 0FFFFFFFh (NULL), the present invention will skip to the offset of the requested bytes and the data for the specified interval is effectively ignored. The present invention will then read the remainder of the data within the particular block and transfer it to the address as requested. A null address will generate an error during a write operation.

Detailed Description Text (100):
The write command transfers data from the system memory 58 and writes it to sequential blocks on the disk drive array. The scatter/gather descriptor count number 2 is ignored by a write command.

Detailed Description Text (122):
FIGS. 22A and 22B are flow diagrams for the PARSE.sub.-- NEW.sub.-- LIST function which is utilized by the present invention to verify the validity of the command list 200 submitted to the disk array controller 112. PARSE.sub.-- NEW.sub.-- LIST is called in step 1100. Control transfers to step 1102 wherein the local processor 122 finds the first byte past the current buffer. Control transfers to step 1104 where the local processor sets data structure pointers to correspond to command list header fields. Control transfers to step 1106 in which the local processor 122 reads the request block counter. Control transfers to step 1108 in which the local processor reads scatter/gather count number 1 fields in the command list header. Control transfers to step 1110 in which the local processor determines the command request size from the request header information. Control transfers to step 1112 in which the local processor determines whether the scatter/gather count number 1 is equal to the size in bytes specified for that particular scatter/gather descriptor. If the scatter/gather count is not equal, control transfers to step 1114 which sets an error code and control is transferred to step 1116 which returns to the calling program with the error code. If the local processor 122 determines in step 1112 that the scatter/gather count number 1 is equal to the size specified, control transfers to step 1118 where the local processor 122 determines whether the second scatter/gather count is non-null. If the second scatter/gather count is null, control transfers to step 1130. If the second scatter/gather count is not null, control transfers to step 1120 where the local processor 122 reads the second scatter/gather count in the request header. Control transfers to step 1122 where the local processor 122 determines the size of the request from the request header. Control transfers to step 1124 where the local processor 122 determines whether the second scatter/gather count is equal to the size of the data. If the second scatter/gather count is not equal, control transfers to step 1126 which sets completion code to false and control is transferred to step 1128 which returns to the calling program. If the second scatter/gather count is equal to the proper size, control transfers to step 1130 where the local processor 122 determines whether the total size of the command list exceeds the maximum 16 Kbyte buffer for each command list and that the size of the command list specified in the command header matches actual size. If the total size of the command list 200 exceeds the allotted buffer size, control transfers to step 1132 in which the local processor 122 sets a completion code equal to FALSE and control transfers to step 1134 which returns to the calling program. If the total size of the command list is equal to or less than the maximum 16 Kbyte buffer, control transfers to step 1136 where the local processor 122 determines if there is a next request following the current command list request. If there is no next request, control transfers to step 1138 wherein the local processor 122 sets the next request to null. Control then transfers to step 1148. If there is a next request, control transfers to step 1140 in which the local processor 122 adds the request offset for the next request to the current request start header. Control transfers to step 1142 where the local processor 122 determines

whether the end of the current request overlaps the calculated address for the next request. If an overlap does occur, control transfers to step 1144 where the local processor 122 sets a completion code equal to FALSE and control transfers to step 1146 which returns control to the calling program. If there is no overlap, control transfers to step 1148 wherein the local processor 122 determines whether there are additional request in the command list 200. If there are additional request in the command list 200, control is transferred to step 1106 and the function continues to PARSE the list until the list has been fully PARSED. If there are no additional requests, control transfers to step 1150 which sets a PARSE completion code equal to TRUE. Control transfers to step 1152 which returns control to the calling program.

Detailed Description Text (125):
The next set of modules address the manner in which the method of the present invention transfers data to and from the host and the disk array. The SG.sub.-- BLOCK module is used to scatter or gather one block of data into or out of the transfer buffers managed by computer system memory. The BMIC.sub.-- XFER module is used to transfer mutiple blocks of data to or from the computer system C utilizing multiple scatter/gather descriptors 208 which are defined in the logical request 204. The BMIC.sub.-- XFER.sub.-- QUEUE module merely queues an incoming transfer request. The BMIC.sub.-- READ module is used by the local processor 122 to read transfer buffers in the disk controller 112 and transfer the data to the host system. Similarly, the BMIC.sub.-- WRITE module is used by the BMIC to write data from the host memory to transfer buffers in the disk array controller 122. The PEEK module sets forth the method by which the present invention reads one byte of data in system memory. Similarly, the POKE module sets forth the method for writing one byte of data to system memory.

Detailed Description Text (126):
6. Scatter/Gather Block Transfer

Detailed Description Text (127):
FIGS. 18A and 18B represent a flow diagram for the SG.sub.-- BLOCK method of transferring data to and from the disk array controller. This method of transferring the data is used in those incidences wherein the command list may contain more than one scatter/gather descriptor block per command request. Operation of the SG.sub.-- BLOCK function begins at step 960. Control transfers to step 962 in which the local processor 122 scans the scatter/gather array for count and determines the address of the start descriptor block. Control transfers to step 964 wherein the local processor 122 determines the host address and the size of the transfer to be made. It should be noted that the host address may include the address of data to be transferred to the host or, the address of data which will be transferred from the host to the disk array controller. Control transfers to step 966 wherein the local processor 122 determines whether calculated address for the beginning of the block is a null address or 0FFFFFFFH. If the calculated address is equal to null, control transfers to step 968 which sets SG.sub.-- TRANSFER flag to FALSE. Control transfers to step 972. If the calculated address is not equal to null, the local processor 122 transfers control to step 970 which sets the SG.sub.-- TRANSFER flag to TRUE. Control transfers to step 972 wherein the local processor 122 determines whether additional data is contained within a current block. It should be noted that an initial null address may be used to indicate a block wherein the data does not occupy the entire block and is offset by a given number of bytes from the beginning of the block.

Detailed Description Text (131):
FIGS. 17A and 17B are flow diagrams of the BMIC.sub.-- TRANSFER function which is utilized to transfer data from the disk array controller through the BMIC to the calling device whether host or some other bus master device. The function is called at step 900. Control transfers to step 902 wherein the local processor determines where the logical blocks for the request start using the parent logical pointer. Control transfers to step 904, wherein the local processor 122 determines whether the logical blocks to be transferred are in a contiguous group. If the logical blocks are not in a contiguous group, control transfers to step 906 which sets the block size to the sector count for that particular contiguous block. Control transfers to step 910. If the local processor 122 determines in step 904 that the blocks to be transferred are in a contiguous group, control transfers to step 908. In step 908, the local processor 122 sets the block size equal to the entire sector count for the transfer. Control transfers to step 910 wherein the local processor 122 obtains the starting block address and pointer to the logical

request. Control transfers to step 912 wherein the local processor 122 determines whether the logical request has only one scatter/gather array. If the logical request has more than one scatter/gather array control transfers to step 938. If the logical request has only one scatter/gather array, a high speed data transfer as follows is effected. Control transfers to step 914 in which the local processor 122 calculates the host start address. Control transfers to step 916 wherein the local processor determines whether the base registers are available. If the base registers are not available, control transfers to step 918 wherein the local processor goes into a wait or poll state and loops back through step 916 until the base registers are available. If the base registers are available, control transfers to step 920, wherein the local processor 122 clears the transfer complete bit and loads the transfer buffer interface data into BMIC 188 channel 0. Control transfers to step 922 wherein the local processor 122 sets BMIC 118 channel 0 buffer address equal to the logical block address for the transfer. Control transfers to step 924 in which the local processor 122 loads the host address and count into BMIC 118 channel 0 base registers. Control transfers to step 926 wherein the local processor 122 loads the output block size of the transfer into the buffer. Control transfers to step 928 in which the local processor 122 determines if the transfer is to be a read or write. If the transfer is a read transfer, control transfers to step 930 which sets the directional bit to transfer data to the EISA master (seen as a write from the disk array controller 112). Control transfers to step 934. If the local processor 122 determines in step 928 that the operation is to be a write, control transfers to step 932 which sets the directional bit to receive data from the EISA master (effectively a read for the disk array controller 112). Control transfers to step 934, wherein the local processor initiates transfer of the 32 bit data, outputs the current channel configuration, channel strobe and increments the host address. Control transfers to step 936 in which the local processor 122 determines if there are more blocks to transfer in the command list. If there are more request blocks to be processed, control transfers to step 916 wherein the local processor 122 continues to process all blocks until the command list 200 is completed. If there are no more request blocks 204 to be processed, control transfers to step 950.

Detailed Description Text (132):
If the local processor 122 determines in step 912 that the logical request has more than one scatter/gather descriptor set 208, control transfers to step 938 which calls the SG.sub.-- BLOCK function which upon sending the block address, block size to the SG.sub.-- BLOCK function, the present invention sends the 32 bit data, the output current channel configuration, channel strobe and increments host address. Control transfers to step 940 in which the local processor 122 determines if there is a second scatter/gather descriptor set 208 for the current request. This would indicate that address information for the read command may be found in two different areas of system memory. If there is no second scatter/gather descriptor set, control transfers to step 944. If there is a second scatter/gather descriptor set, control transfers to step 942 wherein the block address, block size are sent to function SG.sub.-- BLOCK and a transfer of 32 bit data, channel configuration, channel strobe are sent to the host device and the host address is incremented. Control transfers to step 944 wherein the local processor 122 determines if there are additional request blocks to be processed. If there are additional request blocks 204 to be processed control is transferred back to step 938 and the function continues to loop until all request blocks 204 have been processed. If the local processor 122 determines in step 944 that there are no further request blocks 204 to be processed control is transferred to step 946 wherein the local processor 122 advances the BMIC request queue pointer to the next command list 200. Control transfers to step 950 in which the local processor 122 notifies the drive task that the transfer has been completed. Control transfers to step 952 which terminates the operation of this function and returns control to the local processor 122 operating system.

Detailed Description Text (146):
If in step 504, the list header does not have an error or an abort, control transfers to step 514 in which the local processor 122 determines whether the control flag in the command list header is set to notify upon completion. If set to notify upon completion, control transfers to step 516 in which the local processor 122 determines whether the completed request is the last command request in the command list 200. If the command request 204 is the last request in the command list 200, control transfers to step 518 in which the local processor 122 saves the last request as a completed command request address. Control transfers to step 530. If in step 516

it is determined that the request is not the last command request in the command list, control transfers to step 520 and the local processor 122 calls function POST.sub.-- NOTIFICATION to indicate that the particular request in the list has been completed. Control transfers to step 536. If in step 514 it is determined from the command list header control flags that the request is not to notify upon completion, control transfers to step 530. In step 530, the local processor 122 determines whether the command list is complete. If the command list 200 is complete, control transfers to step 532 and the local processor 122 sets the list status in the command list header to list complete. Control transfers to step 534 where the local processor 122 calls function POST.sub.-- NOTIFICATION to send the list complete status to the requesting device. Control transfers to step 536 which frees local memory allocated to the command list headers, command list request and the command list scatter/gather descriptors arrays. Control transfers to step 538 which returns control of the the local processor 122 to the real-time system.

Detailed Description Text (157):
FIG. 15 is a flow diagram of the SENSE.sub.-- CONFIGURATION.sub.-- UNIT function wherein the disk array controller 112 identifies the type of logical unit to the calling device in response to command issued by the calling device. Operation of the SENSE.sub.-- CONFIGURATION function begins at step 800 wherein it is called. Control transfers to step 802 wherein signature address, compatibility port address, operating system, interleave factor, drive count, physical drive count, tolerance mode, and other physical logical parameters are loaded into a data structure in the local memory by the local processor 122. Control transfers to step 804 wherein the local processor 122 roads the RIS data structures for all drives specified in the drive man into local memory. Control transfers to step 806 in which the local processor 122 transfers the data structure from local memory into the transfer buffer. Control transfers to step 808 where the local processor 122 transfers drive array information to the host memory. Control transfers to step 810, where the local processor determines whether there exists a second scatter/gather count field for the particular configuration. If there is no second scatter/gather count, control transfers to step 814 wherein control returns to the local processor 122 operating system. If there is a second scatter/gather count, control transfers to step 812 in which the local processor 122 transfers the second scatter/gather count from the transfer buffer to the host or calling device memory.

Detailed Description Text (158):
18. BMIC Sense Scatter/Gather Structure

Previous Doc Next Doc Go to Doc#

L7: Entry 10 of 10 File: USPT Mar 31, 1992

DOCUMENT-IDENTIFIER: US 5101492 A
**** See image for Certificate of Correction ****
TITLE: Data redundancy and recovery protection

Brief Summary Text (23):
The transfer controller parity circuitry also incorporates four 16 bit parity RAM address registers (0-3) used in conjunction with parity operations. The RAM address registers provide the starting pointers to the transfer buffer memory locations which contain the data blocks to be XOR'd together. Register 0 is assigned to the disk DMA subchannel 3, which, when enabled, is used to manage parity operations. The operation of the parity RAM address registers varies with the number of different blocks that are selected to be XOR'd together and whether the XOR result is to be written back to the transfer buffer or to the parity drive. If four separate block ranges are specified, data will be read from the blocks pointed to by the parity RAM address registers, the data will be XOR'd together and the results will be written to the block addressed by the last parity RAM register or to the parity drive. Should three separate block ranges be selected, the XOR result will be written to the memory location addressed by the parity RAM address register 2. Similarly, when two block ranges are selected, the XOR result will be written to the memory location addressed by parity RAM address register 1.

Detailed Description Text (23):
The processor 20 is preferably an Intel 80386 microprocessor. The processor 20 has its control, address and data lines interfaced to the local processor bus 26. The coprocessor 22 is preferably an Intel 80387 and/or Weitek WTL 3167 numeric coprocessor interfacing with the local processor bus 26 and the processor 20 in the conventional manner. The cache ram 28 is preferably suitable high-speed static random access memory which interfaces with the address and data elements of bus 26 under control of the cache controller 24 to carry out required cache memory operations. The cache controller 24 is preferably an Intel 82385 cache controller configured to operate in two-way set associative master mode. In the preferred embodiment, the components are the 33 MHz versions of the respective units. Address latch circuitry 34 and data transceiver 36 interface the cache controller 24 with the processor 20 and provide a local bus interface between the local processor bus 26 and a host bus 44.

Detailed Description Text (25):
Noncacheable memory address map programmer 30 cooperates with the processor 20 and the noncacheable address memory 32 to map noncacheable memory locations. The noncacheable address memory 32 is utilized to designate areas of system memory that are noncacheable to avoid many types of cache memory incoherency. The bus request logic circuit 42 is utilized by the processor 20 and associated elements to request access to the host bus 44 in situations such as when requested data is not located in the cache memory 28 and access to system memory is required.

Detailed Description Text (45):
A request block 204 is comprised of two parts, a fixed length request header 206 any variable length parameter list 208. The parameters are created as data structures known as scatter/gather (S/G) descriptors which define system memory 58 data transfer addresses. The request header 206 fields contain a link to the next request block 204, the I/O command, space for a return status, a block address and a block count, and a count of the scatter/gather descriptor structure elements for two S/G structures. The request header is a total of 12 bytes

in length.

Detailed Description Text (46):
The scatter/gather descriptor counters are used to designate the number of scatter/gather descriptors 208 which utilized in the particular request. The number of scatter/gather descriptors 208 associated with the request block 204 will vary. Further, if the command is a read command, the request may contain up to two different sets of scatter/gather descriptors. Each scatter/gather descriptor 208 contains a 32 bit buffer length and a 32 bit address. This information is used to determine the system memory data transfer address which will be the source or destination of the data transfer. Unlike the requests blocks 204 in the command list, the scatter/gather descriptors must be contiguous and, if there exists a second scatter/gather descriptor set for a request, it must directly follow the first set of scatter gather descriptors.

Previous Doc Next Doc Go to Doc#

Hit List

First Hit Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 10 of 10 returned.

☐ 1. Document ID: US 20040010674 A1

Using default format because multiple data bases are involved.

L7: Entry 1 of 10 File: PGPB Jan 15, 2004

PGPUB-DOCUMENT-NUMBER: 20040010674
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20040010674 A1

TITLE: Lazy deregistration protocol for a split socket stack

PUBLICATION-DATE: January 15, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
Boyd, William Todd	Poughkeepsie	NY	US
Joseph, Douglas J.	Danbury	CT	US
Recio, Renato John	Austin	TX	US

US-CL-CURRENT: 711/170; 711/154

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

☐ 2. Document ID: US 20030200363 A1

L7: Entry 2 of 10 File: PGPB Oct 23, 2003

PGPUB-DOCUMENT-NUMBER: 20030200363
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030200363 A1

TITLE: Adaptive messaging

PUBLICATION-DATE: October 23, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY
Futral, William T.	Portland	OR	US

US-CL-CURRENT: 710/23

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

☐ 3. Document ID: US 6823437 B2

L7: Entry 3 of 10 File: USPT Nov 23, 2004

US-PAT-NO: 6823437
DOCUMENT-IDENTIFIER: US 6823437 B2

TITLE: Lazy deregistration protocol for a split socket stack

DATE-ISSUED: November 23, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Boyd; William Todd	Poughkeepsie	NY		
Joseph; Douglas J.	Danbury	CT		
Recio; Renato John	Austin	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 10/195180 [PALM]
DATE FILED: July 11, 2002

INT-CL-ISSUED: [07] G06 F 12/00

US-CL-ISSUED: 711/170; 711/203, 710/22, 702/212
US-CL-CURRENT: 711/170; 710/22, 711/203

FIELD-OF-CLASSIFICATION-SEARCH: 711/170, 711/203, 710/22, 702/212
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
5392415	February 1995	Badovinatz et al.	718/100
6034963	March 2000	Minami et al.	370/401
6233244	May 2001	Runaldue et al.	370/412
6662289	December 2003	Ang	711/202
6701420	March 2004	Hamilton et al.	711/170

OTHER PUBLICATIONS

Intel, "Offload Sockets Framework and Sockets Direct Protocol High Level Design", Jun. 2002, p. 2-1, (5-18)-(5-19).*
IBTA, "InfiniBand Architecture Specification vol. 1" Jun. 2001, Release 1.0.a, p. 92-94.*
Dubnicki et al, "Software Support for Virtual Memory-Mapped Comumication", 1996, Proc of IPPS '96, p. 372-381.

ART-UNIT: 2186

PRIMARY-EXAMINER: Padmanabhan; Mano

ASSISTANT-EXAMINER: Baker; Paul A

ATTY-AGENT-FIRM: Yee; Duke W. Salys; Casimer K.

ABSTRACT:

A method, computer program product, and distributed data processing system for lazy deregistration of memory regions. Specifically, the present invention is directed to memory regions that are written to and from by an Integrated Protocol Suite Offload Engine (IPSOE) in accordance with a preferred embodiment of the present invention. A mechanism is provided for lazy deregistration of memory regions once the region is no longer required for a specific data transfer being carried out by the IPSOE. Rather than deregistering a memory region after a data transfer has been carried out, the memory region remains registered for some selected period of time. After that selected period of time passes, the region is then deregistered. If a second data transfer using this memory region occurs while the memory region is still registered, the registration overhead is avoided for this second data transfer. This mechanism reduces the amount of CPU resources required for transferring data by allowing reuse of previously registered memory regions.

36 Claims, 16 Drawing figures

☐ 4. Document ID: US 6615282 B1

L7: Entry 4 of 10 File: USPT Sep 2, 2003

US-PAT-NO: 6615282
DOCUMENT-IDENTIFIER: US 6615282 B1

TITLE: Adaptive messaging

DATE-ISSUED: September 2, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Futral; William T.	Portland	OR		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Intel Corporation	Santa Clara	CA			02

APPL-NO: 09/461237 [PALM]
DATE FILED: December 16, 1999

PARENT-CASE:
This application is a continuation application of Provisional Application Ser. No. 60/135,259, filed on May 21, 1999.

INT-CL-ISSUED: [07] G06 F 13/00

US-CL-ISSUED: 710/1; 710/4, 710/8, 710/20, 710/33, 710/36, 710/39, 710/52
US-CL-CURRENT: 710/1; 710/20, 710/33, 710/36, 710/39, 710/4, 710/52, 710/8

FIELD-OF-CLASSIFICATION-SEARCH: 710/1, 710/4, 710/8, 710/20, 710/33, 710/36, 710/39, 710/52
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
5208810	May 1993	Park	370/230
5475433	December 1995	Jeong	375/240.05
5550957	August 1996	Davidson, Jr. et al.	
5633870	May 1997	Gaytan et al.	
5777624	July 1998	Munson	345/605
5937436	August 1999	Watkins	
6112263	August 2000	Futral	
6125433	September 2000	Horstmann et al.	

ART-UNIT: 2182

PRIMARY-EXAMINER: Gaffin; Jeffrey

ASSISTANT-EXAMINER: Farooq; Mohammad O.

ATTY-AGENT-FIRM: Antonelli, Terry, Stout & Kraus, LLP

ABSTRACT:

In an example embodiment, a data transfer method adaptively transfers data from a host device to a target device across a channel-based interconnect. The method includes determining whether or not the size of the data to be transferred is greater than the maximum payload of a cell for the channel-based interconnect. If the size of the data to be transferred is not greater than the maximum payload, then a single cell is transferred from the host device to the target device which includes all of the data. If the size of the data to be transferred is greater than the maximum payload, then a request message is transferred from the host device to the target device. The request message includes a portion of said data to be transferred and control information indicating that not all of the data to be transferred is included in the request message.

23 Claims, 7 Drawing figures

5. Document ID: US 6505268 B1

L7: Entry 5 of 10 File: USPT Jan 7, 2003

US-PAT-NO: 6505268
DOCUMENT-IDENTIFIER: US 6505268 B1

TITLE: Data distribution in a disk array

DATE-ISSUED: January 7, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schultz; Stephen M.	Houston	TX		
Schmenk; David S.	The Woodlands	TX		
Neufeld; E. David	Tomball	TX		
Grant; David L.	Houston	TX		
Flower; David L.	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corporation	Houston	TX			02

APPL-NO: 09/288399 [PALM]
DATE FILED: April 8, 1999

PARENT-CASE:
This application is a continuation of Ser. No. 08/777,679 Dec. 20, 1996.

INT-CL-ISSUED: [07] G06 F 12/00

US-CL-ISSUED: 711/4; 114/112, 114/111
US-CL-CURRENT: 711/4; 114/111, 114/112

FIELD-OF-CLASSIFICATION-SEARCH: 711/4, 711/114, 711/112, 711/111, 711/170, 711/172, 711/202, 360/69, 360/48
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4215400	July 1980	Denko	
4276595	June 1981	Brereton et al.	
4366512	December 1982	Janak et al.	
4454595	June 1984	Cage	
4590557	May 1986	Lillie	
4608618	August 1986	Sturtevant-Stuart	
4612613	September 1986	Gershenson	
4622602	November 1986	Kutaragi	
4633392	December 1986	Vincent et al.	
4660141	April 1987	Ceccon et al.	
4773004	September 1988	Gershenson	

4805090	February 1989	Coogan
4811279	March 1989	Bean et al.
4817035	March 1989	Timsit
4825403	April 1989	Gershenson
4825404	April 1989	Theus
4843544	June 1989	DuLac et al.
4849929	July 1989	Timsit
4855903	August 1989	Carleton et al.
4885683	December 1989	Coogan
4888691	December 1989	George et al.
4910614	March 1990	Arai et al.
4939598	July 1990	Kulakowski et al.
4951248	August 1990	Lynch
4980850	December 1990	Morgan
5014237	May 1991	Masters
5027313	June 1991	Culley
5148432	September 1992	Gordon et al.
5163131	November 1992	Row et al.
5247633	September 1993	Nissimov et al.
5355453	October 1994	Row et al.

OTHER PUBLICATIONS

Patterson, D., et al., Introduction to Redundant Inexpensive Disk Arrays (RAID), Computer Science Division, Univ. Cal.-Berkley (1989).
Patterson, D., et al., A Case for Redundant Arrays of Inexpensive Disks (RAID), Computer Science Division, Univ. Cal.--Berkley (Dec. 1987).
Schulze, M., Considerations in the Design of a RAID Prototype, Computer Science Division, Univ. Cal.--Berkley (Aug. 1988).
Chen, P., et al., Two Papers on RAID, Computer Science Division, Univ. Cal.--Berkley (Dec. 1988).
Anderson, D., Disk Array Considerations, Imprimis Technology, Inc., date unknown.
Moren, W., Disk Arrays: Performance and Data Availability, Cipricio, Inc., Presented May 24, 1989 at IEEE Systems Design & Networks Conference, Santa Clara, CA.
Schulze, M., et al., How Reliable is RAID?, Computer Science Division, Univ. Cal.--Berkley (reprinted by IEEE 1989).
Katz, R., et al., Industrial Participation in RAID/XPRS, Computer Science Division, Univ. Cal.--Berkley (date believed to be Sep. 2, 1988).
NG, S., Some Design Issues of Disk Arrays, IBM Research Division, (reprinted by IEEE 1989).
Gibson, G., et al., Coding Techniques for Handling Failures in Large Disk Arrays, Computer Science Division, Univ. Cal.--Berkley (Dec. 1988).
Kovsky, S., Pacstor Shows Subsystem with "Fail-Safe" Software, Computer Science News (Apr. 11, 1988).
Data Research Newsletter, Dataquest, Inc. (May 1988).
Promotional Literature on Integra I and III Disk Subsystems, Pacstor, Inc. (1989).
Promotional Literature on Parallel Disk Array Controller, Ciprico, Inc. (date unknown).
Synchronized 51/4 in. Winchester Drives, Electronic Data News (Nov. 26, 1987).
Mokhoff, N., Parallel Disk Assembly Packs 1.5 Gbytes, Runs at 4 Mbytes/s, Electronic Design, pp. 45-46 (Nov. 12, 1987).
Disk Array Forum, Proceedings of Conference held Sep. 18, 1989 in San Jose, CA.
Breaking the Data-Rate Logjam with Arrays of Small Disk Drives, Electronics, pp. 97-99 (Feb. 1989).
Dodge, J., Disk Arrays: Here Come the Disk Drive Sextuplets, Elecronic Business, pp. 126-130 (Nov. 1, 1989).

Promotional Literature on Patrick Henry Disk Array Subsystem, 1776, Inc. (1989).
Chandler, D., Disk Arrays Promise Reliability, Better Access, PC Week, pp. 84, 92 (Oct. 17, 1988).
Lieberman, D., SCSI-2 Controller Board Builds Parallel Disk Drive Arrays, Computer Design, pp. 32, 36 (Apr. 1, 1989).
Williams, T., Disk Array Features 1-Gbyte Fault-Tolerant Storage, Computer Design, p. 33 (Jun. 15, 1988).

ART-UNIT: 2186

PRIMARY-EXAMINER: Peikari; B. James

ATTY-AGENT-FIRM: Akin, Gump, Strauss, Hauer & Feld, LLP

ABSTRACT:

For use with a computer system having an intelligent mass storage disk array subsystem, including a microprocessor controller, a method for the distribution of data within the disk array based upon logical commands issued by the computer system. The disk controller reads a logical command and translates the commands into multiple drive specific commands, including drive physical parameter information such as head, sector and cylinder selection. The calculation of these physical parameters is based upon a number of factors including the operating system installed in the computer system, the type of interleave scheme, if any, specified by the computer system configuration, and disk specific parameters. The physical drive requests are then placed in a queue and executed by the microprocessor controller. The method also encompasses a method for creating a disk array configuration to be loaded on all disks within the array based on existing valid disk array information and configuration information maintained by the computer system.

7 Claims, 30 Drawing figures

6. Document ID: US 5909691 A

L7: Entry 6 of 10 File: USPT Jun 1, 1999

US-PAT-NO: 5909691
DOCUMENT-IDENTIFIER: US 5909691 A

TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

DATE-ISSUED: June 1, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schultz; Stephen M.	Houston	TX		
Schmenk; David S.	The Woodlands	TX		
Neufeld; E. David	Tomball	TX		
Grant; David L.	Houston	TX		
Flower; David L.	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corporation	Houston	TX			02

APPL-NO: 08/777679 [PALM]
DATE FILED: December 20, 1996

PARENT-CASE:
This is a continuation of U.S. application Ser. No. 08/163,011, filed Dec. 7, 1993, now U.S. Pat. No. 5,592,648, which is a divisional of U.S. application Ser. No. 08/145,029, filed Oct. 28, 1993, issued as U.S. Pat. No. 5,440,716; which is a continuation of U.S. application Ser. No. 07/431,748, filed Nov. 3, 1989, now abandoned.

INT-CL-ISSUED: [06] G06 F 12/00

US-CL-ISSUED: 711/4; 711/114, 711/112, 711/111, 395/828, 360/48
US-CL-CURRENT: 711/4; 360/48, 711/111, 711/112, 711/114

FIELD-OF-CLASSIFICATION-SEARCH: 711/4, 711/111, 711/112, 711/170, 711/172, 711/202, 711/114, 395/833, 395/828, 395/894, 360/69, 360/48
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4215400	July 1980	Denko	
4276595	June 1981	Brereton et al.	
4366512	December 1982	Janak et al.	360/48
4454595	June 1984	Cage	
4590557	May 1986	Lillie	395/651
4608618	August 1986	Sturtevant-Stuart	360/135
4612613	September 1986	Gershenson	
4622602	November 1986	Kutaragi	360/48
4633392	December 1986	Vincent et al.	
4660141	April 1987	Ceccon et al.	
4773004	September 1988	Gershenson	
4805090	February 1989	Coogan	
4811279	March 1989	Bean et al.	
4817035	March 1989	Timsit	
4825403	April 1989	Gershenson	
4825404	April 1989	Theus	
4843544	June 1989	DuLac et al.	
4849929	July 1989	Timsit	
4855903	August 1989	Carleton et al.	
4885683	December 1989	Coogan	
4888691	December 1989	George et al.	
4910614	March 1990	Arai et al.	
4939598	July 1990	Kulakowski et al.	360/48

4951248	August 1990	Lynch	
4980850	December 1990	Morgan	
5014237	May 1991	Masters	
5027313	June 1991	Culley	
5148432	September 1992	Gordon et al.	395/182.05
5163131	November 1992	Row et al.	
5247633	September 1993	Nissimov et al.	711/4
5355453	October 1994	Row et al.	

OTHER PUBLICATIONS

Patterson, D., et al., Introduction to Redundant Inexpensive Disk Arrays (RAID), Computer Science Division, Univ. Cal.--Berkley, 1989.
Patterson, D., et al., A Case for Redundant Arrays of Inexpensive Disks (RAID), Computer Science Division, Univ. Cal.--Berkley, Dec. 1987.
Schulze, M., Considerations in the Design of a RAID Prototype, Computer Science Division, Univ. Cal.--Berkley, Aug. 1988.
Chen, P., et al., Two Papers on RAID, Computer Science Division, Univ. Cal.--Berkley, Dec. 1988.
Anderson, D., Disk Array Considerations, Imprimis Technology, Inc., Date Unknown.
Moren, W., Disk Arrays: Performance and Data Availability, Cipricio, Inc., Presented May 24, 1989, at IEEE Systems Design & Networks Conference, Santa Clara, CA.
Schulze, M., et al., How Reliable is RAID?, Computer Sciences Division, Univ. Cal.--Berkley, reprinted by IEEE 1989.
Katz, R., et al., Industrial Participation in RAID/XPRS, Computer Sciences Division, Univ. Cal.-Berkley, Date believed to be Sep. 2, 1988.
Ng, S., Some Design Issues of Disk Arrays, IBM Research Division, reprinted by IEEE 1989.
G. Gibson, et al., Coding Techniques for Handling Failures in Large Disk Arrays, Computer Sciences Division, Univ. Cal.--Berkley, Dec. 1988.
Kovsky, S., Pacstor Shows Subsystem with "Fail-Safe" Software, Computer Science News, Apr. 11, 1988.
Data Research Newsletter, Dataquest, Inc., May 1988.
Promotional Literature on Integra I and III Disk Subsystems, Pacstor, Inc., 1989.
Promotional Literature on Parallel Disk Array Controller, Ciprico, Inc., Date Unknown.
Synchronized 51/4 in. Winchester Drives . . . , Electronic Data News, Nov. 26, 1987.
N. Mokhoff, Parallel Disk Assembly Packs 1.5 Gbytes, Runs at 4 Mbytes/s, Electronic Design, pp. 45-46, Nov. 12, 1987.
Disk Array Forum, Proceedings of Conference held Sep. 18, 1989 in San Jose, CA.
Breaking the Data-Rate Logjam with Arrays of Small Disk Drives, Electronics, pp. 97-99, Feb. 1989.
Dodge, J., Disk Arrays: Here Come the Disk Drive Sextuplets, Electronic Business, pp. 126-130, Nov. 1, 1989.
Promotional Literature on Patrick Henry Disk Array Subsystem, 1776, Inc., 1989.
Chandler, D., Disk Arrays Promise Reliability, Better Access, PC Week, pp. 84, 92, Oct. 17, 1988.
Lieberman, D., SCSI-2 Controller Board Builds Parallel Disk Drive Arrays, Computer Design, pp. 32, 26, Apr. 1, 1989.
Williams, T., Disk Array Features 1-Gbyte Fault-Tolerant Storage, Computer Design, p. 33, Jun. 15, 1988.

ART-UNIT: 272

PRIMARY-EXAMINER: Swann; Tod R.

ASSISTANT-EXAMINER: Peikari; J.

ATTY-AGENT-FIRM: Pravel, Hewitt & Kimball

ABSTRACT:

For use with a computer system having an intelligent mass storage disk array subsystem, including a microprocessor controller, a method for the distribution of data within the disk array based upon logical commands issued by the computer system. The disk controller reads a logical command and translates the commands into multiple drive specific commands, including drive physical parameter information such as head, sector and cylinder selection. The calculation of these physical parameters is based upon a number of factors including the operating system installed in the computer system, the type of interleave scheme, if any, specified by the computer system configuration, and disk specific parameters. The physical drive requests are then placed in a queue and executed by the microprocessor controller. The method also encompasses a method for creating a disk array configuration to be loaded on all disks within the array based on existing valid disk array information and configuration information maintained by the computer system.

11 Claims, 30 Drawing figures

7. Document ID: US 5592648 A

L7: Entry 7 of 10 File: USPT Jan 7, 1997

US-PAT-NO: 5592648
DOCUMENT-IDENTIFIER: US 5592648 A

TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

DATE-ISSUED: January 7, 1997

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schultz; Stephen M.	Houston	TX		
Schmenk; David S.	The Woodlands	TX		
Neufeld; E. David	Tomball	TX		
Grant; David L.	Houston	TX		
Flower; David L.	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corporation	Houston	TX			02

APPL-NO: 08/163011 [PALM]
DATE FILED: December 7, 1993

PARENT-CASE:
This is a divisional, of application Ser. No. 08/145,029, filed Oct. 28, 1993, now U.S. Pat. No. 5,440,716, which is a continuation of application Ser. No. 07/431,748, filed Nov. 3, 1989, now abandoned.

INT-CL-ISSUED: [06] G06 F 7/22

US-CL-ISSUED: 395/441; 395/438, 395/439, 395/825, 360/69, 364/DIG.1
US-CL-CURRENT: 711/114; 360/69, 710/5, 711/111, 711/112

FIELD-OF-CLASSIFICATION-SEARCH: 364/200, 364/900, 360/69MS, 395/425, 395/725, 395/441, 395/438, 395/439
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4215400	July 1980	Denko	395/404
4276595	June 1981	Brereton et al.	395/375
4454595	June 1984	Cage	395/872
4612613	September 1986	Gershenson	395/894
4633392	December 1986	Vincent et al.	395/284
4660141	April 1987	Ceccon et al.	395/829
4773004	September 1988	Gershenson	395/440
4805090	February 1989	Coogan	395/894
4811279	March 1989	Bean et al.	395/306
4817035	March 1989	Timsit	395/185.07
4825403	April 1989	Gershenson	360/52
4825404	April 1989	Theus	395/284
4843544	June 1989	DeLac et al.	395/250
4849929	July 1989	Timsit	395/182.03
4855903	August 1989	Carleton et al.	395/200.05
4885683	December 1989	Coogan	395/183.12
4888691	December 1989	George et al.	395/700
4910614	March 1990	Arai et al.	360/69
4951248	August 1990	Lynch	395/402
4980850	December 1990	Morgan	395/497.03
5014237	May 1991	Masters	395/500
5027313	June 1991	Culley	395/497.03
5163131	November 1992	Row et al.	395/200.01
5355453	October 1994	Row et al.	395/200.09

OTHER PUBLICATIONS

Introduction to Redundant Inexpensive Disk Arrays (RAID), D. Patterson et al., Computer Science Division, Univ. Cal.--Berkley, 1989.
A Case for Redundant Arrays of Inexpensive Disks (RAID), D. Patterson et al., Computer Science Division, Univ. Cal.--Berkley, Dec. 1987.
Considerations in the Design of a RAID Prototype, M. Schulze, Computer Science Div., Univ. Cal.--Berkley, Aug. 1988.
Two Papers on RAID, P. Chen et al., Computer Science Div., Univ. Cal.--Berkley, Dec. 1988.
Disk Array Considerations, D. Anderson et al., Imprimis Technology, Inc., Dat Unknown.
Disk Arrays: Performance and Data Availability, W. Moren, Cipricio, Inc., Presented May 24, 1989 at IEEE Systems Design and Networks Conference, Santa Clara, CA.

How Reliable is RAID?, M. Schulze et al., Computer Sciences Div., Univ. Cal.--Berkley, reprinted by IEEE 1989.
Coding Techniques for Handling Failures in Large Disk Arrays, G. Gibson et al., Computer Sciences Div., Univ. Cal.--Berkley, Dec. 1988.
Industrial Participation in RAID/XPRS, R. Katz et al., Computer Science Div., Univ. Cal.--Berkley, Date believed to be Sep. 2, 1988.
Some Design Issues of Disk Arrays, S. Ng., IBM Research Div., reprinted by IEEE 1989.
Pacstor Shows Subsystem with "Fail-Safe" Software, S. Kovsky, Computer Science News, Apr. 11, 1988.
Dataquest Research Newsletter, Dataquest, Inc., May 1988.
Pacstor, Inc. Promotional Literature on Integra I and III Disk Subsystems, 1989.
Parallel Disk Array Controller Promotional Literature, Ciprico, Inc., Data unknown.
Synchronized 5 1/4 in. Winchester Drives . . . , Electronic Data News, Nov. 26, 1987.
Breaking the Data-Rate Logjam with Arrays of Small Disk Drives, pp. 97099, Electronics, Feb. 1989.
Disk Arrays: Here Come the Disk Drive Sextuplets, J. Dodge, Nov. 1, 1989 Electronic Business, pp. 126-130.
1776, Inc. Promotional Literature for Patrick Henry Disk Array Subsystem, 1989.
Disk Arrays Promise Reliability, Better Access, D. Chandler, Oct. 17, 1988 PC Week, pp. 84, 92.

Disk Array Features 1-Gbyte Fault-Tolerant Storage, T. Williams, Jun. 15, 1988 Computer Design, p. 33.
SCSI-2 Controller Board Builds Parallel Disk Drive Arrays, D. Lieberman, Apr. 1, 1989 Computer Design, pp. 32, 36.
Parallel Disk Assembly Packs 1.5 Gbytes, runs at 4 Mbytes/s, N. Mokhoff, Nov. 12, 1987 Electronic Design, pp. 45-46.
Disk Array Forum, Proceedings of Conference held Sep. 18, 1989 in San Jose CA.

ART-UNIT: 238

PRIMARY-EXAMINER: Swann; Tod R.

ASSISTANT-EXAMINER: Peikari; James

ATTY-AGENT-FIRM: Pravel, Hewitt, Kimball & Krieger

ABSTRACT:

For use with a computer system having an intelligent mass storage disk array subsystem, including a microprocessor controller, a method for the distribution of data within the disk array based upon logical commands issued by the computer system. The disk controller reads a logical command and translates the commands into multiple drive specific commands, including drive physical parameter information such as head, sector and cylinder selection. The calculation of these physical parameters is based upon a number of factors including the operating system installed in the computer system, the type of interleave scheme, if any, specified by the computer system configuration, and disk specific parameters. The physical drive requests are then placed in a queue and executed by the microprocessor controller. The method also encompasses a method for creating a disk array configuration to be loaded on all disks within the array based on existing valid disk array information and configuration information maintained by the computer system.

11 Claims, 30 Drawing figures

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw Desc | Image

☐ 8. Document ID: US 5440716 A

L7: Entry 8 of 10 File: USPT Aug 8, 1995

US-PAT-NO: 5440716
DOCUMENT-IDENTIFIER: US 5440716 A

TITLE: Method for developing physical disk drive specific commands from logical disk access commands for use in a disk array

DATE-ISSUED: August 8, 1995

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schultz; Stephen M.	Houston	TX		
Schmenk; David S.	The Woodlands	TX		
Neufeld; E. David	Tomball	TX		
Grant; David L.	Houston	TX		
Flower; David L.	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corp.	Houston	TX			02

APPL-NO: 08/145029 [PALM]
DATE FILED: October 28, 1993

PARENT-CASE:
This is a continuation of application Ser. No. 07/431,748 filed on Nov. 3, 1989 abandoned.

INT-CL-ISSUED: [06] G06 F 7/22

US-CL-ISSUED: 395/441; 364/251, 364/232.7, 364/295.31, 364/DIG.2, 364/970.5, 360/69, 395/497.01

US-CL-CURRENT: 711/114; 360/69, 711/170

FIELD-OF-CLASSIFICATION-SEARCH: 364/2MSFile, 364/3MSFile, 364/9MSFile, 360/69MSFile, 395/425
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4215400	July 1980	Denko	364/200
4276595	June 1981	Brereton et al.	364/200
4454595	June 1984	Cage	364/900
4612613	September 1986	Gershenson	364/200
4633392	December 1986	Vincent et al.	364/200
4660141	April 1987	Ceccon et al.	364/200
4773004	September 1988	Gershenson	364/200

4805090	February 1989	Coogan	364/200
4811279	March 1989	Bean et al.	364/900
4817035	March 1989	Timsit	364/900
4825403	April 1989	Gershenson	364/900
4825404	April 1989	Theus	364/900
4843544	June 1989	DuLac et al.	364/200
4849929	July 1989	Timsit	364/900
4855903	August 1989	Carleton et al.	364/200
4885683	December 1989	Coogan	364/200
4888691	December 1989	George et al.	364/300
4910614	March 1990	Arai et al.	360/69
4951248	August 1990	Lynch	364/900
4980850	December 1990	Morgan	364/900
5014237	May 1991	Masters et al.	364/900
5027313	June 1991	Culley	364/900

OTHER PUBLICATIONS

Introduction to Redundant Inexpensive Disk Arrays (RAID), D. Patterson et al., Computer Science Division, Univ. Cal.-Berkley, 1989.
A Case for Radundant Arrays of Inexpensive Disks (RAID), D. Patterson et al., Computer Science Division, Univ. Cal.-Berkley, Dec. 1987.
Considerations in the Design of a RAID Prototype, M. Schulze, Computer Science Div., Univ. Cal.-Berkley, Aug. 1988.
Two Papers on RAID, P. Chen et al., Computer Science Div., Univ. Cal.-Berkley, Dec. 1988.
Disk Array Considerations, D. Anderson et al., Imprimis Technology, Inc., Date unknown.
Disk Arrays: Performance and Data Availability, W. Moren, Cipricio, Inc., Presented May 24, 1989 at IEEE Systems Design and networks Conference, Santa Clara, Calif.
How Reliable is RAID?, M. Schulze et al., Computer Sciences Div., Univ. Cal.-Berkley, reprinted by IEEE 1989.
Coding Techniques for Handling Failures in Large Disk Arrays, G. Gibson et al., Computer Sciences Div., Univ. Cal.-Berkley, Dec. 1988.
Industrial participation in RAID/XPRS, R. Katz et al., Computer Science Div., Univ. Cal.-Berkley, Date believed to be Sep. 2, 1988.
Some Design Issues of Disk Arrays, S. Ng., IBM Research Div., reprinted by IEEE 1989.
Pacstor Shows Subsystem With "Fail-Safe" Software, S. Kovsky, Computer Science News, Apr. 11, 1988.
Dataquest Research Newsletter, Dataquest, Inc., May 1988.
Pacstor, Inc. Promotional Literature on Integra I and III Disk Subsystems, 1989.
Parallel Disk Array Controller Promotional Literature, Ciprico, Inc., Date unknown.
Synchronized 5 1/4 in. Winchester Drives . . . , Electronic Data News, Nov. 26, 1987.
Breaking the Data-Rate Logjam With Arrays of Small Disk Drives, pp. 97-99, Electronics, Feb. 1989.
Disk Arrays: Here Come the Disk Drive Sextuplets, J. Dodge, Nov. 1, 1989 Electronic Business, pp. 126-130.
1776, Inc. Promotional Literature for Patrick Henry Disk Array Subsystem, 1989.
Disk Arrays Promise Reliability, Better Access, D. Chandler, Oct. 17, 1988 PC Week, pp. 84, 92.

Disk Array Features 1-Gbyte Fault-Tolerant Storge, T. Williams, Jun. 15, 1988 Computer Design, p. 33.
SCSI-2 Controller Board Builds Parallel Disk Drive Arrays, D. Lieberman, Apr. 1, 1989 Computer Design, pp. 32, 36.
Parallel Disk Assembly packs 1.5 Gbytes, runs at 4 Mbytes/s, N. Mokhoff, Nov. 12, 1987 Electronic Design, pp. 45-46.
Disk Array Forum, Proceedings of Conference held Sep. 18, 1989 in San Jose, Calif.

ART-UNIT: 232

PRIMARY-EXAMINER: Robertson; David L.

ASSISTANT-EXAMINER: Peikari; B. James

ATTY-AGENT-FIRM: Pravel, Hewitt, Kimball & Krieger

ABSTRACT:

For use with a computer system having an intelligent mass storage disk array subsystem, including a microprocessor controller, a method for the distribution of data within the disk array based upon logical commands issued by the computer system. The disk controller reads a logical command and translates the commands into multiple drive specific commands, including drive physical parameter information such as head, sector and cylinder selection. The calculation of these physical parameters is based upon a number of factors including the operating system installed in the computer system, the type of interleave scheme, if any, specified by the computer system configuration, and disk specific parameters. The physical drive requests are then placed in a queue and executed by the microprocessor controller. The method also encompasses a method for creating a disk array configuration to be loaded on all disks within the array based on existing valid disk array information and configuration information maintained by the computer system.

4 Claims, 30 Drawing figures

☐ 9. Document ID: US 5249279 A

L7: Entry 9 of 10 File: USPT Sep 28, 1993

US-PAT-NO: 5249279
DOCUMENT-IDENTIFIER: US 5249279 A

TITLE: Method for controlling disk array operations by receiving logical disk requests and translating the requests to multiple physical disk specific commands

DATE-ISSUED: September 28, 1993

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schmenk; David S.	The Woodlands	TX		
Grant; David L.	Houston	TX		
Schultz; Stephen M.	Houston	TX		
Neufeld; E. David	Tomball	TX		
Flower; David L.	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corporation	Houston	TX			02

APPL-NO: 07/431737 [PALM]
DATE FILED: November 3, 1989

INT-CL-ISSUED: [05] G06 F 7/22

US-CL-ISSUED: 395/425; 395/275, 364/DIG.1, 364/248.1, 364/DIG.2, 364/968.1
US-CL-CURRENT: 710/5

FIELD-OF-CLASSIFICATION-SEARCH: 395/425, 395/275, 395/800, 371/11
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4276595	June 1981	Brereton et al.	364/200
4612613	September 1986	Gershenson et al.	364/200
4773004	September 1988	Gershenson et al.	364/200
4805090	February 1989	Coogan	364/200
4811279	March 1989	Bean et al.	364/900
4825403	April 1989	Gershenson et al.	364/900
4843544	June 1989	DuLac et al.	364/200
4975829	December 1990	Clarey et al.	395/275
4989205	January 1991	Dunphy, Jr. et al.	395/575
5073854	December 1991	Martin et al.	395/425
5097439	March 1992	Patriquin et al.	395/425

OTHER PUBLICATIONS

"82355 Bus Master Interface Controller (BMIC)", 1989 Intel Corporation.
Integra III Block Diagram, Pacstor, Inc., Dec., 1988.
W. Meador, "Disk Array Systems," Maximum Strategy, Inc., 1989 Reprinted by Institute of Electrical and Electronic Engineers.
Ciprico Promotion Literature, for devices disclosed in Computer Design articles.
Lieberman, "VMEbus adapter hosts SCSI-2", Computer Design, Mar. 1989, pp. 98-101.
Lieberman, "Sweep SCSI revision finally approaches standardization", Computer Design, Mar. 1989, pp. 22-25.
Lieberman, SCSI-2 controller board builds parallel disk array, Computer Design, Apr. 1989.
Robert Snively, "Intelligent host adapter directs I/O traffic, freeing up host processor", Electronic Design, Sep. 1984.
Barsky, "Intelligent Controller Trims SCSI Overhead", Electronic Design, Aug. 1989.

ART-UNIT: 232

PRIMARY-EXAMINER: Harrell; Robert B.

ASSISTANT-EXAMINER: Donaghue; L.

ATTY-AGENT-FIRM: Pravel, Hewitt, Kimball & Krieger

ABSTRACT:

A bus master interface command protocol for use with a computer system having an intelligent mass storage disk array subsystem, including a bus master and microprocessor controller. The command protocol permits the computer system to issue disk array commands to the controller at a logical level without having to issue disk specific commands. The disk array subsystem microprocessor controller reads the logical commands, translates the commands into smaller disk specific commands, and queues the disk specific commands for processing. Upon completion of the logical command, the bus master controller asserts control over the computer system bus and manages the transfer of data to or from the computer system memory. The management of the disk array subsystem and the transfer of data is effectively off-loaded from the system processor permitting more efficient use of the processor.

11 Claims, 33 Drawing figures

☐ 10. Document ID: US 5101492 A

L7: Entry 10 of 10 File: USPT Mar 31, 1992

US-PAT-NO: 5101492
DOCUMENT-IDENTIFIER: US 5101492 A
**** See image for Certificate of Correction ****

TITLE: Data redundancy and recovery protection

DATE-ISSUED: March 31, 1992

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Schultz; Stephen M.	Houston	TX		
Schmenk; David S.	The Woodlands	TX		
Flower; David L.	Tomball	TX		
Neufeld; E. David	Tomball	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Compaq Computer Corporation	Houston	TX			02

APPL-NO: 07/431741 [PALM]
DATE FILED: November 3, 1989

INT-CL-ISSUED: [05] G06 F 11/20

US-CL-ISSUED: 395/575; 371/10.2
US-CL-CURRENT: 714/7; 714/710

FIELD-OF-CLASSIFICATION-SEARCH: 371/10.1, 371/10.2, 371/40.1, 364/268.3, 364/268.5, 364/268.9, 364/269.2, 395/575
See application file for complete search history.

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
4276595	June 1989	Brereton et al.	364/200
4454595	June 1984	Cage	364/900
4612613	September 1986	Gershenson	364/200
4773004	September 1988	Gershenson	364/200
4775978	August 1988	Hartness	
4805090	February 1989	Coogan	364/200
4811279	March 1989	Bean et al.	
4817035	March 1989	Timsit	364/900
4825403	April 1989	Gershenson	364/900
4843544	June 1989	DuLac et al.	364/200
4849929	July 1989	Timsit	364/900
4914656	April 1990	Dunphy, Jr. et al.	371/10.2

OTHER PUBLICATIONS

D. Patterson et al, "A Case for Redundant Arrays of Inexpensive Disks (Raid)", ACM Sigmod Conference, Jun. 1-3, 1988.
M. Schulze, "Considerations in the Design of a Raid Prototype," Univ. of California Berkley, Aug. 1988.
C. Gibson, et al., "Coding Techniques for Handling Failures on Large Disk Arrays," Univ. of Cal. Berkley, Dec. 1988.
W. Moren, "Disk Arrays, Performance and Data Availability," IEEE Systems Design Conference, May 24, 1989.
M. Schulze, et al., "How Reliable is Raid?" Univ. of Cal. Berkley, 1989, appointed by IEEE.
D. Patterson, et al., "Introduction to Redundant Arrays of Inexpensive Disk (Raid)," Univ. of Cal. Berkley, 1989, reprinted by IEEE.
P. Chen, et al., "Two Papers on Raids," Univ. of Cal. Berkley, Dec., 1988.
D. Anderson, et al., "Disk Array Considerations" Imprimis Technology Corp. date unknown.
Promotional Literature, 1976 Inc., 1989.
Article Excert "IBM SCSI Card . . . " Infoworld, date unknown.
Informational Literature on Intogra-I, Pacstor Inc., date unknown.
Informational Literature, Fault Tolerance, Pacstore Inc., date unknown.
Informational Literature, Use of DOS V. Unix in Disk Storage Array, Pacstor Inc., date unknown.

Integra-III Block Diagram, Pacstor Inc., 1988.
Promotional Literature for Integra I, Pacstor Inc., 1989.
S. Kousky, "Pacstor Shows Subsystem with `Fail Safe` Software", Apr. 11, 1988, CSN.
D. Britton "Arm Scheduling in Shadowed Disks," Mar, 1989 Compcon, reprinted by IEEE Computer Society.
S. Ng, "Some Design Issues of Disk Arrays," Mar. 1989 Compcon, reprinted by IEEE Computing Society.
W. Meador, "Disk Array Systems," Mar. 1989 Compcon, reprinted IEEE Computing Society.
Dataquist Research News Letter on Pacstor Inc., Integra III, May, 1988.
T. Dodge "Disk Arrays: Here Comes the Disk Drive Sextuples," Nov. 1, 1988, Electronic Business.

"Breake the Data-Rate Logjam with Arrays of Small Disk Drives," Feb. 1989, Electronics.
Promotional Literature of Cipricio Inc. for Parallel Disk Array Controller, date unknown.
D. Chandler, "Disk Arrays Promise Reliability, Better Access," Oct. 17, 1988 PC Week.
Disk Array Forum, Conference Proceedings, Sep. 18, 1989.

T. Williams, "Disk Array Features 1-Gbyte Fault-Tolerant Storage," Jun. 15, 1988, Computer Design.
"Synchronized 51/4--In. Winchester Drives Operates in Parallel and Store 1.5 G Bytes" EDN Nov. 26, 1987.

ART-UNIT: 236

PRIMARY-EXAMINER: Atkinson; Charles E.

ATTY-AGENT-FIRM: Pravel, Gambrell, Hewitt, Kimball & Krieger

ABSTRACT:

A method for detecting the presence of a replacement disk in a fault tolerant, intelligent mass storage disk array subsystem having a microprocessor based controller in a personal computer system and rebuilding the replacement disk independent of the computer system processor. The method calls for the microprocessor controller to run a disk array check at system powerup or at specified intervals to detect the existence of a replacement drive. The microprocessor then builds a series of disk drive commands which attempt to read every sector on the replacement disk. The read commands will return a null data read, indicating that the sector must be restored. The microprocessor controller converts the replacement read commands for all sectors on the replacement disk to write-restore commands. The microprocessor executes the write commands and restores the data to the replacement drive.

10 Claims, 26 Drawing figures

Term	Documents
LIST	416833
LISTS	124716
SCATTER$	0
SCATTER	62821
SCATTERA	2
SCATTERABIL	1
SCATTERABILITY	118
SCATTERABLE	221
SCATTERABLILITY	5
SCATTERABLY	3
SCATTERAD	1
(L6 AND (SCATTER$ NEAR LIST)).PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD.	10

There are more results than shown above. Click here to view the entire set.

Display Format: - Change Format

Previous Page Next Page Go to Doc#